SOFTWARE-BASED METHOD FOR SIMULATION OF MULTIPLE ACCESS NETWORKS

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to a method for accessing to network and, more particularly, to a software-based method for simulation of multiple access networks.

2. Description of Related Art

Conventionally, an Internet protocol is implemented in an existing network (e.g., Ethernet) which is responsible for communicating information. Further, a topology of the network is established to configure communication parameters of a network layer. Network topology is an arrangement of computers, cables, and other branches in a network wherein computers are interconnected for enabling a sharing of resources among them. For obtaining an optimum operation, the topology should be programmed depending on applications. A typical topology falls into one of the following four types:

Bus: It means that each of a plurality of devices (e.g., computers) is coupled to one or more common cables which are the coupled to together.

Star: It means that each of a plurality of computers is coupled to the other computer by cable via a hub.

Ring: It means that a ring is formed by a plurality of computers coupled by cable.

Mesh: It means that any two adjacent computers are coupled together by a unique cable.

However, any above topology is implemented by manually interconnecting cables prior to configuring hardware and/or software. As understood, that manual processing is time consuming and prone to err. Hence, employees have to spend much time on debugging non-critical errors. For example, in the case of interconnecting five computers by adding two computers into the already connected three computers, a disconnection of the established network is possible due to a human error in wiring. In another case of establishing a new Internet protocol, it is necessary to re-program the wiring of network for forming a new network topology for tailoring the needs of experiment or teaching. But above redesign is undesirable because of limited space, inadequate equipment, large scale network layout, high cost, and inflexible network topology.

Therefore, it is desirable to provide a novel method for accessing to network in order to mitigate and/or obviate the aforementioned problems.

SUMMARY OF THE INVENTION

An object of the present invention is to provide a software-based method for simulation of multiple access networks so that a purpose of simulating network topology by software is achieved.

Another object of the present invention is to provide a software-based method for simulation of multiple access networks so that both troubles caused by human errors and time spend by manual wiring are reduced significantly.

Still another object of the present invention is to provide a software-based method for simulation of multiple access networks so that cost of both labor and procurement is reduced significantly.

To achieve the object, the software-based method for simulation of multiple access networks of the present invention comprises a data conversion procedure, a computer-played simulator procedure, a first validation procedure, a second validation procedure, a transmitting simulation frame procedure, a receiving simulation frame procedure, and a software simulation network configuration procedure wherein the data conversion procedure is responsible for converting network configuration information into a data type that is identifiable by a computer and storing the same in a rewritable data storage device which is capable of distributing data into respective computers; the computer-played simulator procedure is responsible for retrieving contents of the rewritable data storage device and following a logic operation based on the contents, so as to act as a simulator for simulating nodes in a network; the first validation procedure is responsible for validating an integrity of network configuration; the second validation procedure is responsible for validating a symmetry of network configuration; the transmitting simulation frame procedure is responsible for transmitting simulation frames to a receiving simulator, wherein the simulation frame is implemented as a data structure capable of communicating among nodes; the receiving simulation frame procedure is responsible for determining whether the received simulation frames are valid or not; and the software simulation network configuration procedure is responsible for establishing a network configuration to be simulated based on above procedures.

Other objects, advantages, and novel features of the invention will become more apparent from the detailed description when taken in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 presents schematically a network topology simulated by software according to the invention;

FIG. 2 presents schematically a relationship among elements of a rewritable data storage device according to the invention;

FIG. 3 is a flow chart illustrating a data conversion procedure according to the invention;

FIG. 4 is a flow chart illustrating a first validation procedure according to the invention;

FIG. 5 is a flow chart illustrating a second validation procedure according to the invention;

FIG. 6 is a flow chart illustrating a transmitting simulation frame procedure according to the invention;

FIG. 7 is a flow chart illustrating a receiving simulation frame procedure according to the invention; and

FIG. 8 is a flow chart illustrating a software simulation network configuration procedure according to the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

In a preferred embodiment of the invention, a broadcast packet switching network (e.g., Ethernet, or the like) is simulated. The invention can simulate one or more networks complying with broadcast medium on one or more computers installed with a protocol (e.g., TCP/IP (Transport

Control Protocol/Internet Protocol) for transferring simulation frames to computers involved in the simulation) by using software. Also, simulation frames are transferred over the simulated network. The invention takes advantage of the installed protocol to simulate a frame transfer mechanism. Further, the invention can define and simulate a real network topology by software for transmitting/receiving simulation frames. As a result, a simulation of multiple access networks is effected.

A computer employed by the invention comprises a rewritable data storage device for storing data about the network topology and associated parameters. Data may be object-oriented, relation-oriented, or the other as long as it can correctly represent the network topology.

With reference to FIG. 1, there is shown a schematic drawing of a network topology simulated by software according to the invention. The topology is implemented as a bus topology and comprises links L_i (i from 1 to 2) implemented as a transferring medium for transferring simulation frames; nodes N_i (i from 1 to 6) in the network; and network interfaces I_i (i being one of 11, 12, 2, 31, 32, 4, 5, and 6). One interface I_i is permitted to couple to at most one link L_i in compliance with the only linking requirement about the network interface I_i. Further, one network interface I_i is permitted to belong to at most one node N_i in compliance with the only subordination requirement about network interface I_i. Furthermore, one link L_i may be coupled to at least one network interface I_i or nothing. Moreover, one node N_i can have at least one network interface I_i or nothing. In the embodiment, network interface I_i is implemented as a network adapter, while it is appreciated by those skilled in the art that it may be a modem or port without departing from the scope and spirit of the invention.

A data conversion procedure is employed to convert network topology of FIG. 1 into data which can be stored in computer. A flow chart illustrating the data conversion procedure is shown in FIG. 3. First, a user defines a simulation (S301) as shown in FIG. 1. Then analyze a configuration relationship among nodes N_i, links L_i, and network interfaces I_i (S302). In FIG. 1, there are provided two links L_i, six nodes N_i, and eight network interfaces I_i. Next, mathematical sets are employed to represent nodes N_i, links L_i, and network interfaces I_i so as to form a data type identifiable by the computer (S303). In the case a set theorem of mathematics is employed to modualize the relationship of links L_i, nodes N_i, and network interfaces I_i shown in FIG. 1. Hence, links L_i and nodes N_i may be represented as follows:

$L_1=\{I_{11}, I_2, I_{31}, I_4\}$,

$L_2=\{I_{32}, I_5, I_6\}$,

$N_1=\{I_{11}, I_{12}\}$,

$N_2=\{I_2\}$,

$N_3=\{I_{31}, I_{32}\}$,

$N_4=\{I_4\}$,

$N_5=\{I_5\}$, and

$N_6=\{I_6\}$.

The data type obtained from above simulation is stored in the rewritable data storage device (S304). Also, the rewritable data storage device can be provided in respective computers. In the case, the link L_i is defined to have a unique ID (e.g., link_id) for uniquely identifying a specific link L_i by the rewritable data storage device. Note that the link_id may be a string of characters. The network interface I_i has a unique ID (e.g., interface_id) for identifying a specific network interface I_i. Note that the interface_id is a value having a fixed bit representing an address of medium access control layer of the network interface I_i. Also, any two or more network interfaces I_i coupled to the same link L_i are not allowed to have the same interface_id in compliance with the only identification requirement about the network interface I_i on link L_i. Similarly, any two or more network interfaces I_i coupled to the same node N_i are not allowed to have the same interface_id in compliance with the only identification requirement about the network interface I_i on node N_i.

With reference to FIG. 2, there is shown a schematic diagram depicting possible relationship between elements of the rewritable data storage device. As shown, each block represents an element and each line connecting two blocks represents a relationship therebetween. The relationship can be "acting as", "having", "belong to", or "coupled to".

The invention employs a computer-played simulator procedure to cause a computer to retrieve contents of the rewritable data storage device and follow a logic operation based on the contents, so as to act as a simulator S_i (i=1, 2, 3, or 4). Simulator S_i is typically a process of the operating system and acts as nodes N_i. One simulator S_i can act as one or more nodes N_i. Also, one node N_i can act as one or more simulators S_i. Preferably, for the sake of design, one simulator S_i acts as one node N_i having the lowest complexity.

In the embodiment, simulator S_i defines a protocol as UDP (User Datagram Protocol) of TCP/IP. A process in each operating system is implemented as a simulator S_i for monitoring an UDP port of IP address. In a specific simulation, simulator S_i can only act as one node N_i. Also, each simulator S_i has to transmit or receive simulation frames via UDP.

Based on the data conversion procedure, the rewritable data storage device may distribute data into respective computers. Hence, it is important to validate whether the distributed data complies with the original topology. FIG. 4 is a flow chart illustrating a first validation procedure for validating an integrity of the network configuration. In the first validation procedure, R, L, N, I and S represent the rewritable data storage device, all links L_i represented by R, all nodes N_i represented by R, all network interfaces I_i represented by R, and all simulators S_i executed by R, respectively. Further, data of the rewritable data storage device is distributed into k computers and represented as R_1, R_2, R_3,..., and R_k, wherein $R_1=\{L_1, N_1, I_1, S_1\}$, $R_2=\{L_2, N_2, I_2, S_2\}$,...,$R_k=\{L_k, N_k, I_k, S_k\}$.

First, let $L=L_1 \cup L_2 \cup L_3 \ldots \cup L_k$ (S401); $N=N_1 \cup N_2 \cup N_3 \ldots \cup N_k$ (S402); and $I=I_1 \cup I_2 \cup I_3 \ldots \cup I_k$ (S403). Then, it is validated whether any two links L_x and L_y in the rewritable data storage device satisfy the equation $(L_x \cap L_y) = \emptyset$ (S404). If it is true, it means that it complies with the only linking requirement about the network interface I_i. Next, it is validated whether any two nodes N_x and N_y therein satisfy the equation $(N_x \cap N_y) = \emptyset$ (S405). If it is true, it means that it complies with the only subordination requirement about the network interface I_i. Note that the first validation procedure is also applicable to a validation of data in a non-distributive rewritable data storage device as long as k has a value of one.

In addition to validate the integrity of topology, it is still required to validate a symmetry of nodes N_i in the rewritable data storage device in order to determine whether the simulator S_i participating in the simulation can obtain address information about the simulator S_i with the help of the rewritable data storage device. With reference to FIG. 5, there is shown a flow chart illustrating a second validation procedure according to the invention. The second validation procedure is substantially the same as the first one. For example, the second validation procedure defines k simulators S_i (e.g., S_1, S_2, S_3,..., and S_k) participating a simulation.

As shown, nodes N_i are first defined to be simulated by a certain simulator S_i as $N(S_i)$, i=1, 2, 3,...,k (S501), so as to form a set. Next, let $N(S_i)=\{N_1, N_2, N_3,...,N_m\}$ (S502). Then, $L(N_j)$ is defined as a set of all links L_i coupled to the node N_i where j=1,2,3,...,m (S503). Next, $L(S_i)$ is defined as a set of all links L_i related to the simulator S_i (S504), that is, $L(S_i)=L(N_1)\cup L(N_2)\cup L(N_3)\cup \ldots L(N_m)$. Then, any two simulators S_x and S_y are defined to be symmetric if they satisfy $L(S_x)\cap L(S_y)\neq\emptyset$ (S505). Hence, the simulator S_x obtains the complete information of the simulator S_y by the rewritable data storage device in the protocol (e.g., address information, UDP port, or the like). Similarly, the simulator S_y obtains the complete information of the simulator S_x in the protocol by the rewritable data storage device.

A simulation of the invention is completed if the first and second validation procedures are fulfilled. Next, a procedure of simulating communication between nodes N_i is performed based on the completed simulation. First, in the simulation frame, there are defined an ID (e.g., sender_id) for recording a network interface I_i sending the simulation frame, an ID (e.g., destination_id) for recording a network interface I_i desired to receive the network interface I_i, and an ID (e.g., bearing_link_id) for identifying a link L_i which transports the simulation frame.

10067227.020702

With reference to FIG. 6, there is shown a flow chart illustrating a transmitting simulation frame procedure according to the invention. The simulation frame is defined as a data structure of nodes N_i in communication. The transmitting simulation frame procedure is responsible for sending simulation frames to a simulator S_i for receiving. First, it is assumed that the simulation frame to be sent is F, I_m is a network interface I_i for sending F, L_m is a link L_i coupled to I_m, and I_n is a destination network interface I_i of F (S601). Further, the interface_id of I_m is filled into the sender_id field of F (S602), the interface_id of I_n is filled into the destination_id field of F (S603), and the link_id of L_m is filled into the bearing_link_id field of F (S604). Note that the order of steps S602, S603, and S604 may be altered. Next, a protocol is called for taking the simulation frame as a payload of protocol (S605). Finally, the simulation frame is transmitted to the simulator S_i conforming to the simulation (S606).

At this time, simulator S_i has to determine whether to process or discard a received simulation frame. In response, a receiving simulation frame procedure has to be performed. With reference to FIG. 7, there is shown a flow chart illustrating the receiving simulation frame procedure. In the receiving simulation frame procedure, I(N) is defined as a set comprising all network interfaces owned by nodes. Also, N(S) and L(S) have to satisfy steps S501, S502, S503, and S504 shown in FIG. 5. Further, I[N(S)] is defined as a set comprising all network interfaces I_i of all nodes N_i simulated by the simulator S_i. The simulator S_i first receives the simulation frame (S701), and next retrieves an address of the network interface I_i to be transmitted from the simulation frame (S702). Such an address is the destination_id stored in the simulation frame as indicated in the transmitting simulation frame procedure. Further, link L_i information used between two network interfaces I_i (S703) is retrieved, i.e., the bearing_link_id stored in the simulation frame. Then, the network interface I_i (or link information) stored in the simulation frame is compared with that in the simulator S_i (S704). The simulation frame can be accepted by the simulator S_i only after the following conditions are satisfied: there exists an interface_id of the network interface I_i in I[N(S)] of simulator S_i wherein the interface_id is equal to the destination_id of the simulation frame, and the link_id of the link L_i coupled to the network interface I_i is equal to the bearing_link_id of the simulation frame; or the destination_id of the simulator S_i is equal to a specific value (e.g., broadcasting address of LAN (Local Area Network)) validated by a certain simulator Si, and the bearing_link_id of simulator is in the set of L(S) (S705).

With reference to FIG. 8, there is shown a flow chart illustrating a software simulation network configuration procedure for establishing a network configuration based on above procedures. First, a network configuration is programmed to be simulated (S801). For the sake of programming, it is preferably to schematically draw a network topology to be simulated as that shown in FIG. 1. Next, the programmed network configuration is converted to be simulated into a data type identifiable by the computer by performing the data conversion procedure (S802). Then, a required data field for each element is added in the rewritable data storage device (e.g., data of ID). Thereafter, the number of simulators S_i and the number of nodes N_i to be simulated by simulator S_i are determined (S803). Next, the rewritable data storage device is divided and distributed into each of the simulators S_i for being stored (S804). Finally, it is ascertained that each simulator S_i is capable of transmitting and receiving simulation frames. Also, each simulator S_i is capable of recognizing structure information of the other one (S805).

Although the present invention has been explained in relation to its preferred embodiment, it is to be understood that many other possible modifications and variations can be made without departing from the spirit and scope of the invention as hereinafter claimed.